Answers to The Second Most Asked Question

One of the things that's been on my list to write about since before I really had a list is the dilemma of how one predicts the performance of an application in a production environment that doesn't match its test environment—a situation that usually arises because it is impractical or impossible to do performance testing in the actual production environment. The reason I haven't taken it up before now is that I didn't think that what I had to say would take enough words to be worth publishing formally. But yesterday I got yet another e-mail asking for advice on this topic.

Scott Barber

I can tell you that this is probably the question that I have been asked the second most number of times since about four years ago when I was silly enough to start putting my e-mail address on a Web site and in my sig on the forums I frequent. (In case you are wondering, the number one question is "What tool should I use?" or some variation.)

As I answered the e-mail, I realized that my "quick" answer had turned into a page, and I began to wonder if maybe I'd been wrong about not having much to say on the topic. So I spent way more time than I probably should have scanning my e-mail archive folders and old forum posts and found that I'd written plenty more than an article's worth of responses already.

Thinking (mistakenly) that it would be quicker and easier to consolidate what I'd already written than to start over, I compiled the following list of related questions and answers about performance testing with limited or no access to the production environment. The questions and answers are in chronological order, and I have removed any potentially identifying information from the original questions (and lightly edited in a few places for brevity or clarity), but the only changes that have been made to my responses are typographical or grammatical.

The Q & A

Q: [Winter, '02] When distributing results of performance tests executed in the test lab, people always ask me what the results will equate to in the production environment. Obviously, the environments are very different and it is not a straightforward task to try and predict performance in one environment based on results from another (or am I just being slow?). Does anybody have any strategies for approaching this problem?

A: No, it's complicated at best. Personally, I think predicting the performance of one environment based on the performance of another is pure "black magic." Having said that, here are my thoughts on what you could do:

1. Get a good capacity planning book (probably one by Daniel Menascé[1]).

2. Consider applying Software Performance Engineering (SPE), as Connie Smith[2] suggests.

3. Do what I do. Tune it the best you can in the test environment. Make your best educated guess, then tell [the client] that you need at least two hours of testing in the prod. environment before you go live. During those two hours, execute a single one-hour test designed to tell you if your prediction is even close and what the actual performance is for a specific case. Based on that test, [the client] can decide whether to leave it in prod. or to pull it back out.

For what it's worth, I NEVER make any committal statements about one environment based on another. The closest to committal I'm willing to go is: "Yup, it should be faster/better, but only if everything works like you say it will. I wouldn't bet on it if I were you."

Q: [Spring, '03] We are planning to set up a lab for performance testing and are trying to decide on performance [requirements] for the application, but we don't know how to account for the difference between lab and production environments. I would like to know how one can make use of the results obtained in the lab environment to [assess the performance of] the production environment.

A: You have two options:

1. Make your best estimate, and find a way to validate it in production before it's too late to make changes.

2. Apply SPE on the front end, validate the model in the lab environment with your load generation tools in the middle and then do a full capacity planning and scalability exercise on the back end with the validated SPE model and known production hardware.

Q: [Spring, '03] [A comment by Roland Stens[3]:] *On a side note, isn't it odd that we do all this work to be precise with our scenarios, load simulation, measurements, etc., to simply do some pie-in-the-sky extrapolations to give the client an idea on how his application might do in the production environment? This always leaves a bad taste in my mouth.*

A: That is how I (almost) always manage to get my tests run in the production environment before go-live day: "If you're not going to let me validate our guess-timations on the production system, then there's no point in my scripting anything else—what we have is already more realistic than the environment you've got me testing in."

Q: [Spring, '04] I know that it is not good

Scott Barber is the CTO at PerfTestPlus. His specialty is context-driven performance testing and analysis for distributed multi-user systems. Contact him at sbarber@perftestplus.com.

to extrapolate results. But my client is insisting that I do exactly that for my report... [technical details omitted].

A: There is no formula or methodology that is any more accurate than a guess [if you are starting from load simulations]—in fact, guessing is probably the most accurate way to predict [in this scenario]. Were I in your shoes, I'd tell [the client] to get me more hardware/licenses so I could add more users. If they said no, then I would tell them flatly that I could not provide [the number] they are asking for. Personally, I would much rather be removed from a project/get fired than be forced into providing information that I can virtually guarantee is wrong. In my book, that is unethical, and I refuse to compromise my ethics for employment.

Q: [Winter, '06] Our test manager, my immediate report, is of the opinion that if we do not have a full blown load test environment that is identically sized to the production environment with every application [that will be sharing the production environment] performing a background load for [our application], we are just wasting our time and money by creating and executing load tests.

Up until a few months ago, we had testing applications in isolation because the effort of testing everything together is rather challenging and may result in no testing at all or a failure to meet specified deadlines. Would it be possible for you to comment on this topic?

A: Sure, but it sounds like you're going to need more than just my comments to achieve your goal.

There is some validity to your manager's position IF the only reasons you are performance testing is to either determine specific performance characteristics of the application in production or to determine SLA compliance in production.

HOWEVER, even if one or both of these are true, developing and debugging your load to be generated is much smarter to do in a development or test environment. The only things that I can think of more likely to cause devastating side effects if performed directly on the production system than developing and debugging performance scripts are security and reliability/fail-over testing.

Now, assuming that these are not your only reasons for conducting a performance test, and you'd actually like to find and fix bottlenecks, performance testing in virtually ANY environment is valuable. The drawback to non-production-like test environments is that there are certain aspects of the production environment that you won't account for, so while you certainly can determine if the application's performance is bad, you simply cannot determine if it's "good enough."

Most organizations have no way to extrapolate performance results from a test environment.

For all practical intents and purposes, most organizations have no way to extrapolate performance results from a test environment to predict production performance. Even the best performance testers in the field refer to this kind of extrapolation as "black magic" unless they've hired Smith, Menascé or one of their "trained lieutenants" to do it for them.

A better reason, in my book, to not do all of your performance testing in the production environment is that starting to test in the most complex environment simply obscures bottlenecks—making them more difficult to pinpoint and resolve. I recommend starting in the most simple, limited, isolated environment possible and adding components only when the performance of the previous configuration is at least understood. This progressive approach saves a fantastic amount of time and effort every time you detect a performance issue.

Now, all this is not to say that I am against testing in the production environment or an environment that mirrors production. What I am saying, however, is that should account for something like 10 percent of your total test time...which often means that it can actually be done in production during scheduled maintenance periods, thus saving the money of building a test environment that mirrors production AND mitigating the risk of performance testing in production.

When it's all said and done, even application modeling master Smith, who can predict the performance of an application being deployed to a new environment down to the millisecond after the application has been modeled and validated in a controlled test environment, writes that you still need to validate the predictions by generating real load on the application.

I mean, think about it: How is extrapolation, no matter how scientific or accurate in a controlled, sterile environment, going to account for someone installing a hard drive with a slower seek speed than planned for in the model, or for a mouse-chewed network cable, or for that one configuration setting that didn't get changed to allow the application to make use of the hyper-threading technology on the new hardware that the model had accounted for?

I guess it's just another variation on what is quickly and unexpectedly becoming my catchphrase for 2006: If you want to have any confidence about what performance will be like in production, you're just going to have to "investigate in test and validate in production." ⊠

ENDNOTES

1. Co-author of "Performance by Design: Computer Capacity Planning By Example," Prentice Hall PTR, 2004.
2. Co-author of "Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software," Addison-Wesley Professional, 2001.
3. Performance testing consultant.